Final Report Contract No. GR-501720 NPS-52KI0061A

United States
Naval Postgraduate School

COMPUTER-AIDED DESIGN OF LINEAR NETWORKS IN THE FREQUENCY DOMAIN

by

Donald E. Kirk

11 June 1970

COMPUTER-AIDED DESIGN OF LINEAR NETWORKS IN THE FREQUENCY DOMAIN

Final Report NPS-52KI0061A

by

Donald E. Kirk

NAVAL POSTGRADUATE SCHOOL
Monterey, California

11 June 1970
Contract Number GR-501720

This work was performed for the Jet Propulsion Laboratory, California Institute of Technology, as sponsored by the National Aeronautics and Space Administration under Contract NAS7-100

NAVAL POSTGRADUATE SCHOOL
Monterey, California

Rear Admiral R. W. McNitt, USN
Superintendent

R. F. Rinehart
Academic Dean

ABSTRACT:

The design of linear networks to satisfy frequency domain performance specifications is formulated as a problem in nonlinear programming. Three optimization algorithms, pattern search, gradient projection, and the Fletcher-Powell method, are applied in conjunction with the network analysis program CALAHAN to the solution of the nonlinear programming problem. Examples which illustrate the range of application of the design programs are presented.

This investigation was supported by:

Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, California 91103
Purchase Order No. GR-501720

Donald E. Kirk

Donald E. Kirk
Associate Professor of
Electrical Engineering

Approved by:

Charles H. Rothauge

Charles H. Rothauge
Chairman
Department of Electrical
Engineering

C. E. Menneken

C. E. Menneken
Dean of Research
Administration

TABLE OF CONTENTS

1. INTRODUCTION

The recent development of several digital computer programs for the analysis of electrical circuits has added a new dimension to procedures for designing networks. These network analysis programs have the capability to calculate time and frequency response data from a topological description of an electric circuit. The response data generated can be used by the designer to alter the network parameters and configuration until a satisfactory design is achieved. If this approach to network design is followed, it is important to have a digital computer operating mode with minimal turn-around time so that the designer can observe the effects of parameters changes and proceed to an acceptable design in an efficient manner.

An alternative approach is to use a digital computer program to make design decisions. When this is done, a network analysis program supplies performance data which is used to alter the network parameters in a systematic way. The parameters are adjusted to minimize a performance measure; the performance measure (or objective function) is a function which indicates the deviation of the network response from the desired response. Minimization of the performance measure (or optimization) can be carried out by any of several computational algorithms.

Regardless of whether the design is performed by repeated analysis, or by optimization, the computer-aided approach offers several advantages over classical design procedures, e.g. Darlington synthesis, Brune cycle, etc. In computer-aided design procedures, parameter constraints, nonlinearities, and parasitic effects can be included. The design of a network of specified complexity whose response approximates a desired response can also be carried out. In addition, computations can be performed using the physical design parameters, such as transistor base width, or a film thickness in an integrated circuit, as

the variables. References 1 - 4 provide a good summary of the state of the art of computer-aided network design.

The investigation reported here has been limited to the design of linear networks in the frequency domain. Three optimization algorithms have been used in combination with the network analysis program CALAHAN to solve several example problems.

2. NETWORK DESIGN AS A NONLINEAR PROGRAMMING PROBLEM

The basis for computer-aided network design by the use of numerical optimization procedures is the formulation of the design as a problem in nonlinear (or mathematical) programming. A nonlinear programming problem is one in which a function J (the objective function) of r variables $p_1, p_2, \ldots, p_r$ is to be minimized (or maximized) subject to constraining relations of the form

$$g_i(p_1, \ldots, p_r) \geq 0 \quad , \; i = 1,2,\ldots,m \quad . \tag{1}$$

To simplify the notation, vector-matrix symbols will henceforth be used, that is, $\underset{\sim}{p} \triangleq \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_r \end{bmatrix}$, and (1) can be written

$$g_i(\underset{\sim}{p}) \geq 0 \quad , \; i = 1,2,\ldots,m \quad , \tag{1a}$$

or, simply

$$\underset{\sim}{g}(\underset{\sim}{p}) \geq \underset{\sim}{0} \quad , \tag{1b}$$

where

$$\underset{\sim}{g}(\underset{\sim}{p}) \triangleq \begin{bmatrix} g_1(\underset{\sim}{p}) \\ g_2(\underset{\sim}{p}) \\ \vdots \\ g_m(\underset{\sim}{p}) \end{bmatrix}$$

The set of points which satisfy all of the constraints given by (1) is called the feasible set. The constraints in (1) that are satisfied with the equality holding are said to be active.

The feasible set is assumed to be a closed, bounded, and convex set, and the objective function is assumed to have continuous second partial derivatives with respect to the components of $\underset{\sim}{p}$.

In the network design application, the objective function J provides a measure of the deviation of the network frequency response from the desired frequency response, and the variables are a set of r network parameters $p_1, p_2, \ldots, p_r$. In general, the performance measure will be assumed to have the form

$$J(\underset{\sim}{p}) = \sum_{i=1}^{N} h(G(f_i), G_D(f_i), f_i) \tag{2}$$

where f_i(Hz), i = 1,2,...,N, is a set of frequencies for which the desired response is specified, $G_D(f_i)$ is the desired response at $f = f_i$, and $G(f_i)$ is the response of the network at the frequency f_i. The scalar function h satisfies the following properties:

1. $h \geq 0$, for all $G(f_i), G_D(f_i), f_i$, $i = 1,2,\ldots,N$ (3a)

2. $h = 0$, only if $G(f_i) = G_D(f_i)$ (3b)

In the example problems discussed subsequently, the performance measured used was

$$J(p) = \sum_{i=1}^{N} w_i[20 \log_{10} |G_D(f_i)| - 20 \log_{10} |G(f_i)|]^2 \tag{4}$$

where the w_i are positive numbers called weighting factors. By inspection of (4) it is observed that the larger a particular w_i is, the larger the value of J for a given response deviation; hence, by adjusting the w_i's the actual response can be made to conform more closely to the desired response at certain frequencies at the cost of larger deviations at other frequencies.

In this investigation the constraints considered were restricted primarily to the form

$$a_i \leq p_i \leq b_i \ , \quad i = 1,2,\ldots,r \ , \tag{5}$$

where a_i and b_i represent the lower and upper bounds for the ith parameter. These constraints can be put into the form of (1) by defining

$$
\begin{aligned}
g_1(\underset{\sim}{p}) &= p_1 - a_1 \geq 0 \\
g_2(\underset{\sim}{p}) &= -p_1 + b_1 \geq 0 \\
&\vdots \\
g_{2r-1}(\underset{\sim}{p}) &= p_r - a_r \geq 0 \\
g_{2r}(\underset{\sim}{p}) &= -p_r + b_r \geq 0
\end{aligned}
\tag{6}
$$

For some of the optimization algorithms considered, extension to other types of constraints is a routine matter; this will become evident in the subsequent discussion.

3. SOLUTION OF NETWORK DESIGN PROBLEMS

To design a network by using the procedures described here the designer must first select the configuration of the network. Some investigators have reported the incorporation of logic into design programs for "growing" network elements, but this investigation did not include this possibility.

After specifying the network configuration, a performance measure and any weighting factors must be selected. The desired frequency response characteristic must be specified either in the form of a table of values, or as an analytical expression that can be included in the program.

Once these preliminary steps in the design procedure have been completed, and the necessary data has been provided for the combined analysis-optimization program, the design parameters are adjusted by the computer program until the performance measure is minimized. The network analysis program provides frequency response data to the optimization program which then adjusts the variable parameters in a systematic way until the minimum is found. The nature of the data that must be provided by the analysis program depends on the optimization algorithm being used. Generally, the network analysis program will be required to provide either values of J only, or values of J and the first partial derivatives of J with respect to the elements of $\underset{\sim}{p}$.

A flow chart of the computational procedure is shown in Figure 1. It should be noted that although this flow chart and the procedure outlined are for frequency-domain design, they apply also to design in the time domain by replacing the work "frequency" by the word "time". The time response, however, would be calculated within the network analysis program in a different manner than the frequency response.

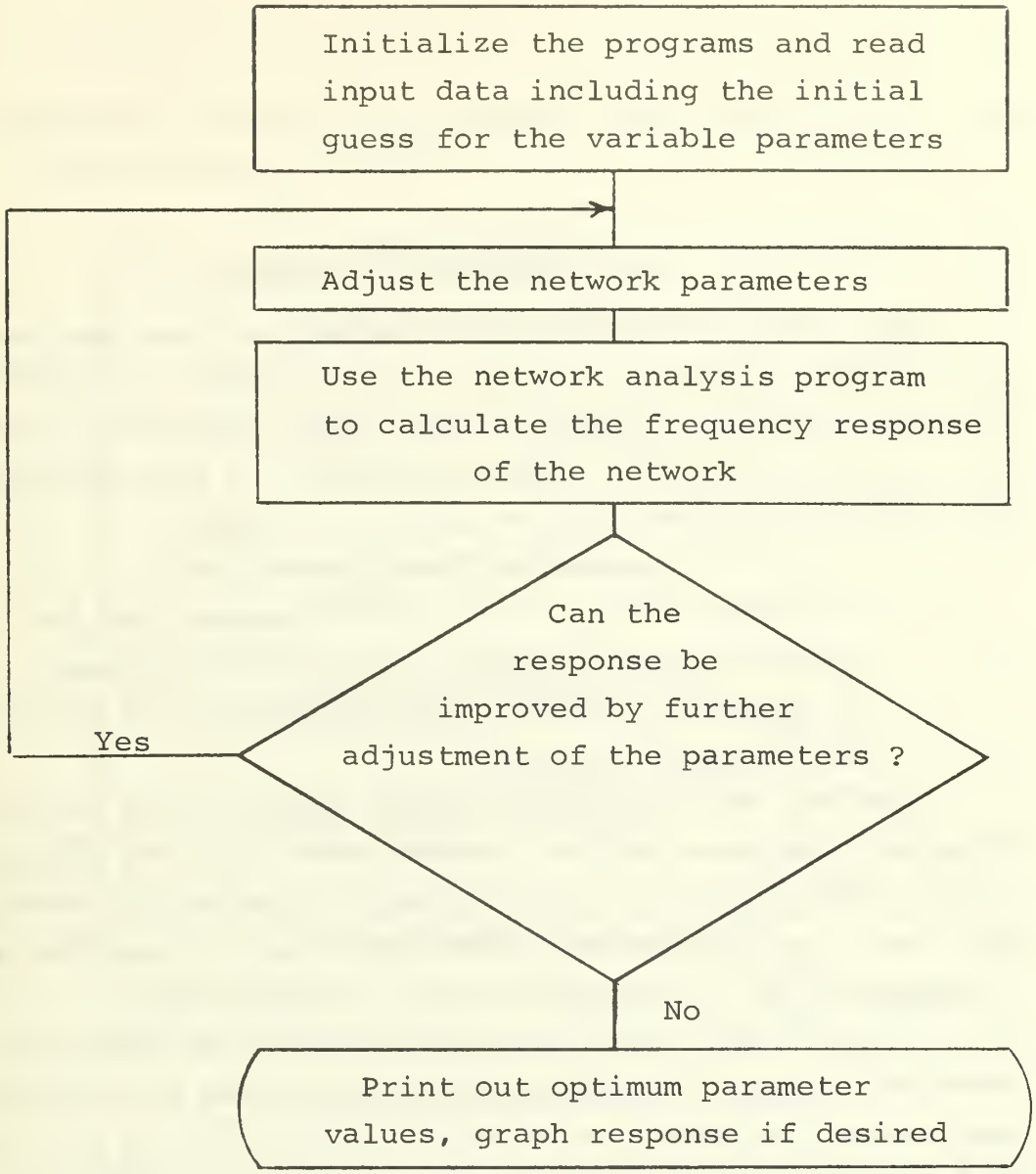


Figure 1 Flowchart of the computational procedure

4. THE NETWORK ANALYSIS PROGRAM

There are many network analysis programs that have been written. Most of these programs are available from the authors or from the sponsoring agencies. A few of the more widely known analysis programs are CALAHAN, SCEPTRE, ECAP, CORNAP, and NASAP. In this investigation the program CALAHAN has been used exclusively because:

1. It is well documented and hence easy to use.
2. It provides both time and frequency response data for linear networks (including networks with dependent sources).
3. The program is compatible with the IBM 360/67 in use at the Naval Postgraduate School.

Experience has indicated the CALAHAN is adequate for use with the optimization programs employed; however, it is clear that modifications could be made which would make the program operate more efficiently when used in an iterative mode. This comment will be amplified after reviewing the use of the CALAHAN network analysis program.

To use CALAHAN, a topological description of the network and numerical values for its parameters must be provided as input data to the program. The network to be analyzed is considered to be in the form of a two-port black box as shown in Figure 2. One of the user options in CALAHAN is the selection of the network function of interest;

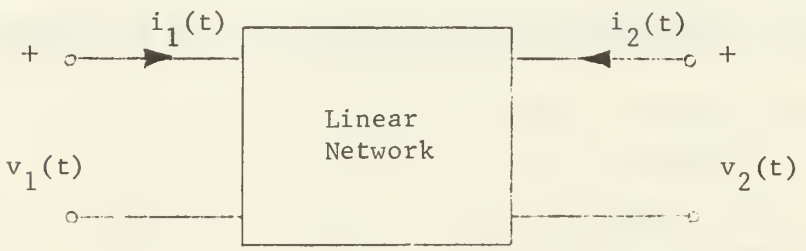


Figure 2 Two-port network

the choice is made by specifying the input parameter KEY 1. The alternatives are shown in Table 1.

KEY 1	Network Function	Symbolic
1	Voltage transfer function	$\frac{V_2}{V_1} \Big\vert_{I_2 = 0}$
2	Open circuit driving point impedance	$\frac{V_1}{I_1} \Big\vert_{I_2 = 0}$
3	Open circuit transfer impedance	$\frac{V_2}{I_1} \Big\vert_{I_2 = 0}$
4	Short circuit driving point impedance	$\frac{I_1}{V_1} \Big\vert_{V_2 = 0}$
5	Short circuit transfer admittance	$\frac{I_2}{V_1} \Big\vert_{V_2 = 0}$
6	Current transfer function	$\frac{I_2}{I_1} \Big\vert_{V_2 = 0}$

TABLE 1. Network function options

The other input data that must be provided to the modified version of CALAHAN is:

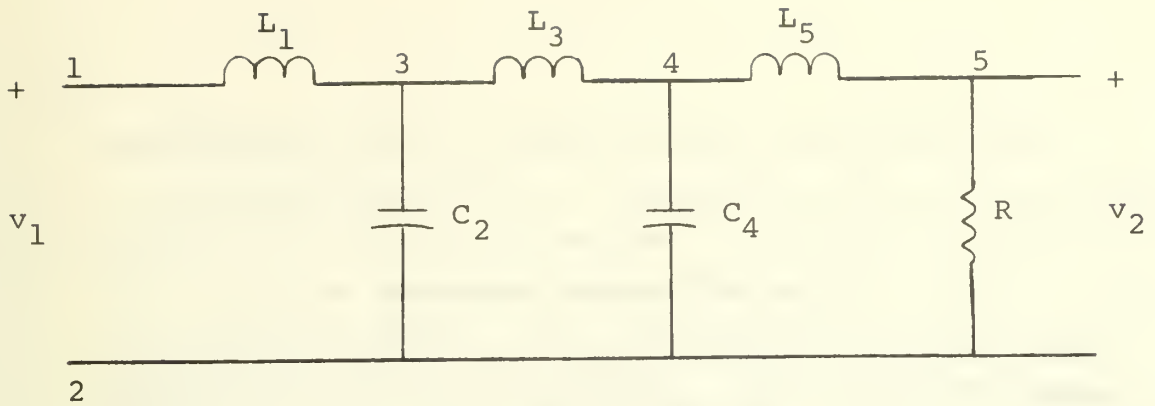
1. The number of passive (RLC) elements.
2. The number of controlled sources (current sources that are voltage controlled).
3. The number of nodes.
4. The positive input node.
5. The negative input node.
6. The positive output node.
7. The negative output node.
8. KEY 1

9. The location, element type, and value of each of the network parameters.

10. The frequency response data.

Items 1 through 9 are placed on a single data card according to the format (I2, 1X). Item 9 is provided by supplying one data card for each element; this data card contains the node numbers at the ends of the element, the element type, and the element value according to the format (2(I2, 1X), A1, 1X, F10.0). The information for Item 10 is provided on two cards: the first card, which is read according to the format (I1, 1X, I3), contains a 1 or 2 in column 1 depending on whether a linear or logarithmic frequency scale is desired, and columns 3 through 5 contain either the total number of frequency intervals (for a linear scale), or the number of points per decade (for a log frequency scale); the second card, which is read according to the format (2F10.0), contains the smallest and largest frequencies. Figure 3 illustrates a circuit diagram and the input data to compute the frequency response $V_2(f)/V_1(f)$ for f = 0.150 Hz. to f = 0.250 at intervals of Δf = .001 Hz. The "output" data that result in this example are the values of $V_2(f)/V_1(f)$ for f = 0.150, 0.151, ..., 0.250. The unmodified version of CALAHAN provides additional data, including the poles and zeros of the network function, and plots of magnitude and phase versus frequency; however, since this information is not desired when the analysis program is used in an iterative mode, it has been eliminated from the program.

As mentioned previously, the use of CALAHAN with an iterative optimization algorithm is somewhat inefficient. The reason for this is that each time CALAHAN is called, the tree-generating subroutine begins as if a new network were being analyzed. As a result, a disproportionate amount of time is spent generating information that could be computed once and stored.



L_1 = 1.5451 H. C_2 = 1.6944 F.

L_3 = 1.3820 H. C_4 = 0.8944 F.

L_5 = 0.3090 H. R = 1.0 Ω

(a) The circuit diagram

	Columns 1 - 10	Columns 11 - 20	Columns 21 - 30 ...
Card # 9	0.150	0.250	
Card # 8	1 100		
Card # 7	05 02 R 1.0		
Card # 6	04 05 L 0.3090		
Card # 5	04 02 C 0.8944		
Card # 4	03 04 L 1.3820		
Card # 3	03 02 C 1.6944		
Card # 2	01 03 L 1.5451		
Card # 1	06 00 05 01 02 05 02 01		

(b) The input data

Figure 3

5. THE OPTIMIZATION ALGORITHMS

Three optimization algorithms have been combined with CALAHAN (in separate programs) to perform optimal design of linear networks in the frequency domain. A description of these optimization algorithms follows.

5.1 The Gradient Projection Method

The gradient projection algorithm, developed by J. B. Rosen (see reference 5) determines the minimum of a function of several variables which are constrained to lie in a closed and bounded convex region R in the parameter space P. R is defined as the interior of a convex region bounded by a set of linear constraints. A detailed exposition of Rosen's method is given in reference 5 and also in reference 6. Here, a brief description of the important features of the algorithm will be given.

The algorithm is, as the name implies, a gradient search, but with the capability of solving problems in which the variables are constrained. The version of Rosen's algorithm that has been used in this investigation assumes that the constraints are linear, that is,

$$\underset{\sim}{N}_{\ell}^{T}\underset{\sim}{p} - \underset{\sim}{v}_{\ell} \geq \underset{\sim}{0} \quad , \tag{7}$$

where $\underset{\sim}{N}_{\ell}$ is a known r x ℓ matrix, $\underset{\sim}{p}$ is the vector of r variable parameters, $\underset{\sim}{v}_{\ell}$ is a known vector of ℓ constants, and the superscript T denotes matrix transposition. By selecting $\ell = 2r$ and

$$\underset{\sim}{N}_{\ell} = \left[\begin{array}{c|c} \underset{\sim}{I}_r & -\underset{\sim}{I}_r \end{array}\right] , \quad \underset{\sim}{v}_{\ell} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_r \\ -b_1 \\ -b_2 \\ \vdots \\ -b_r \end{bmatrix}$$

the formulation in (7) yields the special case given in (6).

The performance measure J is minimized subject to the constraints given by (7). The strategy employed in the search procedure is to move in the direction of steepest descent (the negative gradient direction) in the parameter space so long as the function values decrease and none of the constraints are violated. If a move in the negative gradient direction would cause one or more of the constraints to be violated, the negative gradient is projected onto the constraint boundary at the point where the constraint violation would occur. This projection of the negative gradient is accomplished by a q x q projection matrix $\underset{\sim}{P}_q$; q is the number of constraints in (7) that are <u>active</u> (satisfied with the equality).

The projection matrix is formed from the columns of the matrix $\underset{\sim}{N}_\ell$ which correspond to active constraints. The active constraints can be written as

$$\underset{\sim}{N}_q^T \underset{\sim}{p} - \underset{\sim}{v}_q = \underset{\sim}{0} \quad , \tag{8}$$

where $\underset{\sim}{v}_q$ is formed by selecting those entries of $\underset{\sim}{v}_\ell$ which correspond to active constraints. The projection matrix is given by

$$\underset{\sim}{P}_q = \underset{\sim}{I} - \underset{\sim}{N}_q[\underset{\sim}{N}_q^T \underset{\sim}{N}_q]^{-1} \underset{\sim}{N}_q^T \tag{9}$$

The algorithm is carried out in such a manner that $\underset{\sim}{P}_q$ can be computed by using recurrence equations that do not require inversion of the matrix $[\underset{\sim}{N}_q^T \underset{\sim}{N}_q]$; this feature provides a tremendous computational saving. If the current best point does not lie on the boundary, but within R, then $\underset{\sim}{P}_q$ is the identity matrix.

If $\underset{\sim}{p}^{(i)}$ is the current best point the next parameter vector is given by

$$\underset{\sim}{p}^{(i+1)} = \underset{\sim}{p}^{(i)} + \tau \underset{\sim}{P}_q[-\partial J(\underset{\sim}{p}^{(i)})/\partial \underset{\sim}{p}]/\|\partial J(\underset{\sim}{p}^{(i)})/\partial \underset{\sim}{p}\| \tag{10}$$

where τ (the step size) is calculated by using an interpolation procedure or a single variable search.

To illustrate how the gradient projection algorithm operates, a simple example is shown in Figure 4. In this example the constraints are given by

$$0 \leq p_1 \leq 1$$

$$0 \leq p_2 \leq 1 \quad , \tag{11}$$

or

$$\begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{bmatrix}^T \underset{\sim}{p} - \begin{bmatrix} 0 \\ 0 \\ -1 \\ -1 \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{11a}$$

Several equal-value contours of the function J are shown in Figure 4 and R is the region bounded by these constraints. Notice that R is a convex set because every point on a straight line drawn between any two points in R is also in R.

If $\underset{\sim}{p}^{(o)}$ is the first trial point, then $\underset{\sim}{z}^{(o)}$ is the unit vector in the negative gradient direction and a move is made to the boundary point $\underset{\sim}{p}^{(1)}$ on the line H_4. At $\underset{\sim}{p}^{(1)}$, moving in the negative gradient direction would violate the constraints, so a move is made in the direction of the vector $\underset{\sim}{z}^{(1)}$ (a unit vector in the direction of the projection of the negative gradient at $\underset{\sim}{p}^{(1)}$ onto the line H_4). Moving in this direction yields $\underset{\sim}{p}^{(2)}$ where further motion in the direction of $\underset{\sim}{z}^{(1)}$ would violate the constraint $p_2 \leq 1$. Repeating the steps used earlier provides the information that no improvement can be made by moving from $\underset{\sim}{p}^{(2)}$; hence, the iterative procedure terminates.

This algorithm has been coded in FORTRAN IV by the principal investigator; in reference 7 a description is given of the combined operation of CALAHAN and the gradient projection algorithm.

Two features of the CALAHAN - gradient projection program are of special interest.

1. The gradient projection algorithm requires not only values of J, but also the partial derivatives of J from which the

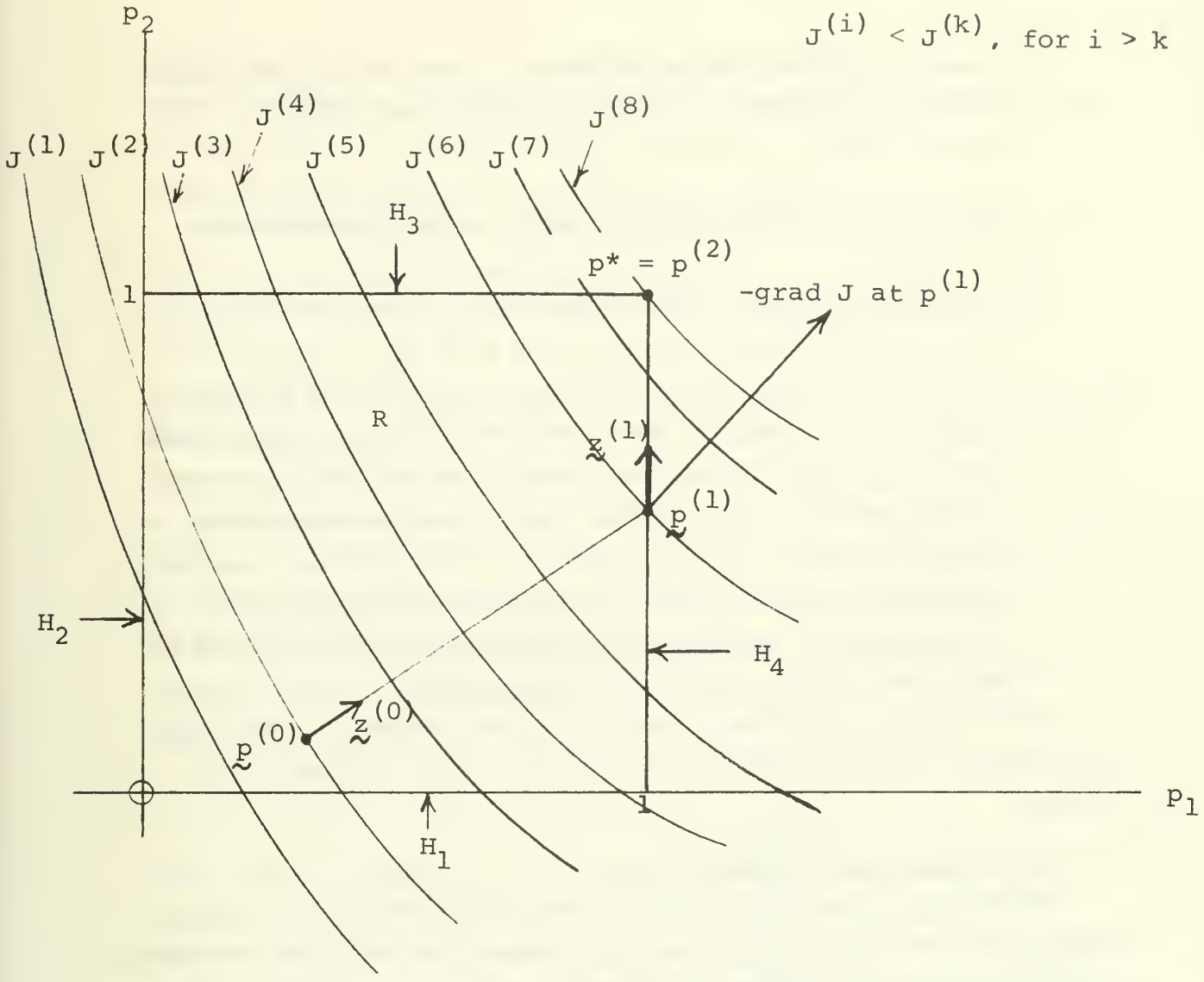


Figure 4 Geometric interpretation of the gradient projection algorithm

negative gradient vector is formed. These partial derivatives were approximated by a finite difference approximation, that is

$$\frac{\partial J}{\partial p_j}(\underset{\sim}{p}) \approx \frac{J(p_1,p_2,\ldots,p_j+\Delta,\ldots,p_r) - J(p_1,p_2,\ldots,p_j,\ldots,p_r)}{\Delta}$$

Using this approach, (r+1) network analyses are required to evaluate the gradient vector at the point $\underset{\sim}{p}$.

2. The interpolation scheme for finding the step size τ suggested in reference 5 requires additional evaluations of the gradient vector. Because the gradient calculation is costly in terms of the number of calculations, an alternative approach was adopted in which only the values of J were needed. In addition to proving more efficient in this application, this single variable search procedure alleviated difficulties believed to be caused by the occurrence of local minima.

Reference 7, in addition to describing the combined operation of the programs, also provides several examples which illustrate applications.

5.2 The Pattern Search Method

Unlike the gradient projection method, which requires the first partial derivatives of J with respect to each of the variable parameters, the pattern search algorithm, reference 8, requires only values of the function J. Perturbations in the elements of the parameter vector $\underset{\sim}{p}$ are used to determine the direction of a change in $\underset{\sim}{p}$ which decreases the value of J. The perturbation data, which provides local information, is then used to suggest larger changes in $\underset{\sim}{p}$ which should further decrease J. These larger changes in $\underset{\sim}{p}$ are called pattern moves. The algorithm alternates perturbation moves (or local explorations) with pattern moves. Each time a successful pattern move occurs, the step size of the following pattern move is increased; thus, continued successful pattern moves cause future moves to become larger and larger. When a pattern move eventually fails, the pattern is destroyed and the

algorithm attempts to begin generating a new pattern. The search terminates when an exploration move cannot decrease the function, and the perturbation step size has been reduced to a value below a pre-selected minimum.

Figure 5 illustrates a simple two-parameter problem whose solution indicates the steps followed by the pattern search algorithm. The figure shows the two-dimensional parameter space and several equal-value contours of J.

At the initial trial point, $\underset{\sim}{b}^{(1)}$, the function J is evaluated, and this value is denoted by $J(\underset{\sim}{b}^{(1)})$. The parameter p_1 is perturbed by an amount $+ \Delta p_1$ and the function value increases, indicating a failure, so p_1 is perturbed by $- \Delta p_1$ and the function value decreases (a success) from its value at $\underset{\sim}{b}^{(1)}$. Perturbation of p_2 by an amount Δp_2 further decreases the function and yields $\underset{\sim}{b}^{(2)}$ as the current best point. Using the rationale that further perturbations would likely result in the same pattern of successes, the next move is to the point $\underset{\sim}{t}^{(1)}$, where

$$\underset{\sim}{t}^{(1)} = \underset{\sim}{b}^{(1)} + 2[\underset{\sim}{b}^{(2)} - \underset{\sim}{b}^{(1)}] \quad . \tag{13}$$

Notice that the function is not evaluated at $\underset{\sim}{t}^{(1)}$ unless all perturbations about $\underset{\sim}{t}^{(1)}$ produce values of J greater than $J(\underset{\sim}{b}^{(2)})$ -- which does not occur for the geometry of this example. The iterative procedure continues in this manner with perturbation cycles and pattern moves alternating. The pattern moves are determined by using the equation

$$\underset{\sim}{t}^{(i)} = \underset{\sim}{b}^{(i)} + 2[\underset{\sim}{b}^{(i+1)} - \underset{\sim}{b}^{(i)}] \tag{14}$$

Several additional moves are also shown in Figure 5.

As mentioned previously, the pattern moves continue to increase in length as long as successes occur. If a failure occurs on a pattern move [all perturbations about $\underset{\sim}{t}^{(i)}$ produce values of J larger than $J(\underset{\sim}{b}^{(i+1)})$ and $J(\underset{\sim}{t}^{(i)}) > J(\underset{\sim}{b}^{(i+1)})$], then the pattern is destroyed.

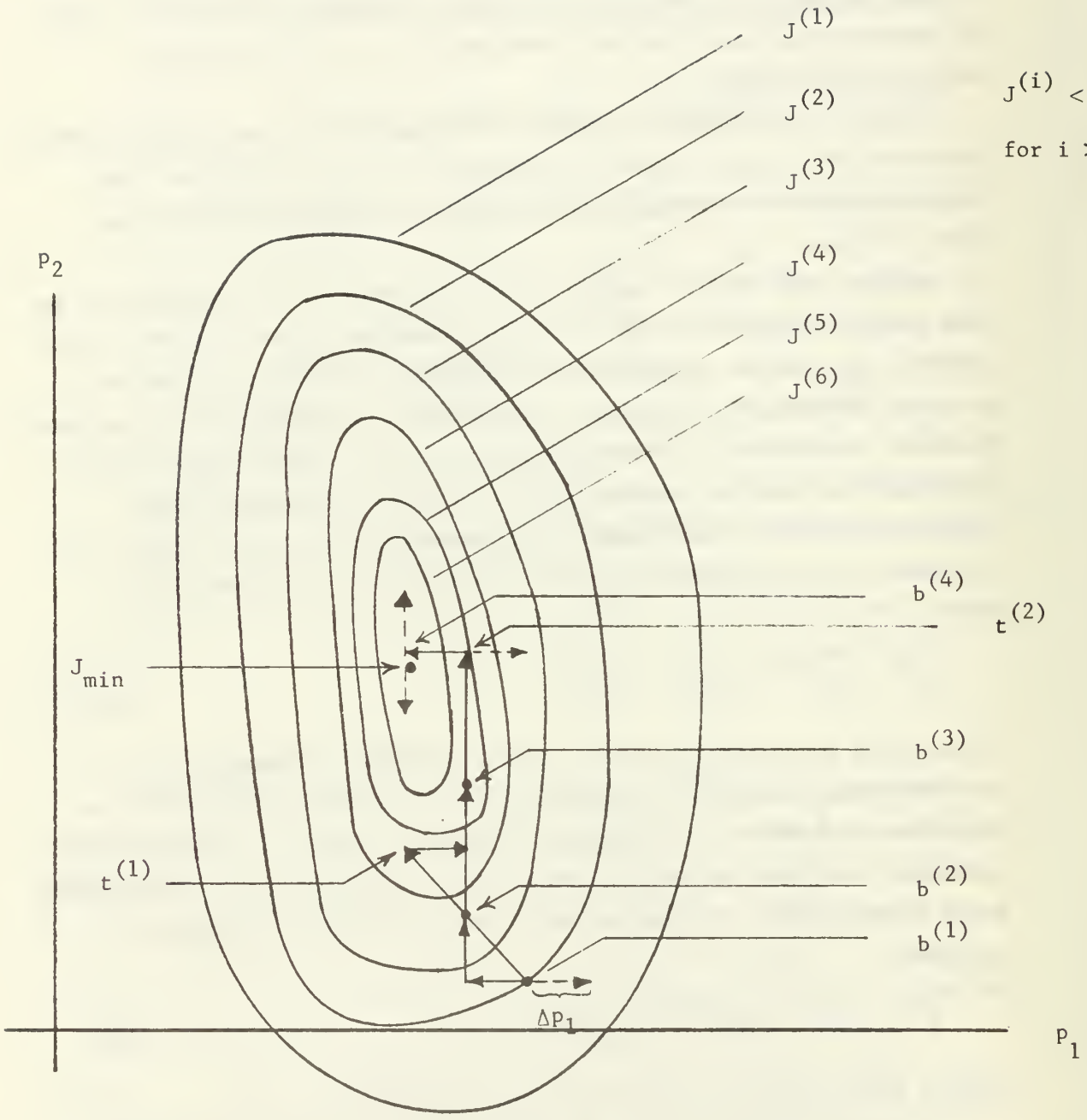


Figure 5 Geometric interpretation of the pattern search algorithm

When it becomes impossible to make successful perturbation moves from a current best point, the perturbation step size is decreased by a specified amount. When the perturbation step size has decreased below a pre-specified level, the iterative procedure terminates.

The pattern search method is available as subroutine DIRECT in the Naval Postgraduate School Computer Facility FORTRAN Library. In reference 9 a description is given of the combined operation of CALAHAN and the pattern search algorithm. The version of pattern search reported in reference 9 was modified to incorporate constraints of the form

$$a_i \leq p_i \leq b_i \quad , \quad i = 1,2,\ldots,r \tag{15}$$

in the algorithm. Reference 9 also presents several network design examples.

5.3 The Fletcher-Powell Method

The Fletcher-Powell method, reference 10, is basically a gradient method; however, unlike other gradient techniques, the convergence of the algorithm becomes more rapid as the minimum is approached. The reason for this is that the gradient vectors calculated in the iterative procedure are used to generate an estimate of the matrix of second partial derivatives. As the procedure evolves, the estimated matrix of the second partial derivatives approaches the actual matrix of second partial derivatives and the convergence of the algorithm becomes quadratic.

The motivation for this approach is provided by considering the Taylor series expansion of J about the point $\underset{\sim}{p}^{(i)}$ which includes terms of up to second order

$$J\left(\underset{\sim}{p}^{(i+1)}\right) = J\left(\underset{\sim}{p}^{(i)}\right) + \left[\frac{\partial J}{\partial \underset{\sim}{p}}\left(\underset{\sim}{p}^{(i)}\right)\right]^T \left[\underset{\sim}{p}^{(i+1)} - \underset{\sim}{p}^{(i)}\right] + \frac{1}{2}\left[\underset{\sim}{p}^{(i+1)} - \underset{\sim}{p}^{(i)}\right]^T \frac{\partial^2 J}{\partial \underset{\sim}{p}^2}\left(\underset{\sim}{p}^{(i)}\right)\left[\underset{\sim}{p}^{(i+1)} - \underset{\sim}{p}^{(i)}\right] , \tag{16}$$

where

$$\frac{\partial J}{\partial \underset{\sim}{p}} \triangleq \left[\frac{\partial J}{\partial p_1} \quad \frac{\partial J}{\partial p_2} \quad \cdots \quad \frac{\partial J}{\partial p_r} \right]^T$$

denotes the gradient of J with respect to $\underset{\sim}{p}$, $\frac{\partial J}{\partial \underset{\sim}{p}} (\underset{\sim}{p}^{(i)})$ is the gradient vector at the point $\underset{\sim}{p}^{(i)}$ and $\frac{\partial^2 J}{\partial \underset{\sim}{p}^2} (\underset{\sim}{p}^{(i)})$ is the matrix of second partial derivatives

$$\begin{bmatrix} \frac{\partial^2 J}{\partial p_1^2} & \frac{\partial^2 J}{\partial p_1 \partial p_2} & \cdots & \frac{\partial^2 J}{\partial p_1 \partial p_r} \\ \frac{\partial^2 J}{\partial p_2 \partial p_1} & \frac{\partial^2 J}{\partial p_2^2} & \cdots & \frac{\partial^2 J}{\partial p_2 \partial p_r} \\ \vdots & \vdots & & \vdots \\ \frac{\partial^2 J}{\partial p_r \partial p_1} & \frac{\partial^2 J}{\partial p_r \partial p_2} & \cdots & \frac{\partial^2 J}{\partial p_r^2} \end{bmatrix}$$

evaluated at the point $\underset{\sim}{p}^{(i)}$.

The point $\underset{\sim}{p}^{(i)}$ is known (initially it is a guess) and the point $\underset{\sim}{p}^{(i+1)}$ is to be found. The gradient of J with respect to $\underset{\sim}{p}^{(i+1)}$ is

$$\frac{\partial J}{\partial \underset{\sim}{p}} \left(\underset{\sim}{p}^{(i+1)} \right) = \frac{\partial J}{\partial \underset{\sim}{p}} \left(\underset{\sim}{p}^{(i)} \right) + \frac{\partial^2 J}{\partial \underset{\sim}{p}^2} \left(\underset{\sim}{p}^{(i)} \right) \left[\underset{\sim}{p}^{(i+1)} - \underset{\sim}{p}^{(i)} \right] , \quad (17)$$

and if $\underset{\sim}{p}^{(i+1)}$ is to be the minimizing point, it is necessary that

$$\frac{\partial J}{\partial \underset{\sim}{p}} \left(\underset{\sim}{p}^{(i+1)} \right) = \underset{\sim}{0} . \quad (18)$$

Hence,

$$\underset{\sim}{p}^{(i+1)} = \underset{\sim}{p}^{(i)} - \left[\frac{\partial^2 J}{\partial \underset{\sim}{p}^2}\left(\underset{\sim}{p}^{(i)}\right)\right]^{-1} \frac{\partial J}{\partial \underset{\sim}{p}}\left(\underset{\sim}{p}^{(i)}\right) \triangleq \underset{\sim}{p}^{(i)} - \underset{\sim}{Q}^{-1} \text{ grad } J^{(i)} \tag{19}$$

This equation suggests that to find $\underset{\sim}{p}^{(i+1)}$ the change in $\underset{\sim}{p}$ should not be in the negative gradient direction (this would be the case if the matrix of second partials were the identity matrix), rather a deflected gradient should be used. If the matrix of second partials $\underset{\sim}{Q}$ (the Hessian matrix) were known, Equation (19) could be used as the basis for an algorithm having quadratic convergence. Unfortunately, when an explicit relationship for J in terms of $\underset{\sim}{p}$ is not known, as will be the case in computer-aided network design applications, the computation of $\partial^2 J/\partial \underset{\sim}{p}^2$ is very difficult, and attempting to find the matrix of second partial derivatives by perturbations often leads to inaccurate results.

In the Fletcher-Powell Method, the gradients themselves are used to obtain an estimate of $\underset{\sim}{Q}$. For a function J that is quadratic in $\underset{\sim}{p}$ (equation (16) is exact), the algorithm converges in r iterations, where r is the dimension of $\underset{\sim}{p}$.

The iteration equation used is

$$\underset{\sim}{p}^{(i+1)} = \underset{\sim}{p}^{(i)} - \tau \underset{\sim}{H}^{(i)} \text{ grad } J^{(i)}, \tag{20}$$

where τ is the step size which is determined by performing a single variable search. The single variable search in the subroutine being used involves first an extrapolative phase, during which the value of τ which makes $J^{(i+1)}$ the smallest along the line in the deflected gradient direction is bracketed, and an interpolation phase, during which the minimizing value for τ is determined.

The Fletcher-Powell Method as described is capable of solving unconstrained minimization problems; however, in the class of problems of interest here the variable parameters will be required to satisfy constraints. To allow application of the Fletcher-Powell Method some way of incorporating the constraints must be found. One way of

handling constraints is to use penalty functions; the performance measure J is altered by some additional terms which depend on the constraints. Intuitively speaking, the terms in the altered performance measure, J_a, that correspond to the constraints should be large if the constraint boundaries are being approached and small if the parameters are within the region where the constraints are satisfied.

To be specific, assume that the constraints are of the form

$$g_i(\underset{\sim}{p}) \geq 0, \quad i = 1,2,\ldots,m. \tag{21}$$

As usual, it is desired to minimize J. The method used is to form the altered performance measure (devised by Fiacco and McCormick, reference 11)

$$J_a(\underset{\sim}{p}) = J(\underset{\sim}{p}) + R \sum_{i=1}^{m} \frac{1}{g_i(\underset{\sim}{p})} \quad . \tag{22}$$

R is a positive constant.

If the initial point lies in the interior of the feasible region, i.e., where

$$g_i(\underset{\sim}{p}) > 0, \quad i = 1,2,\ldots,m \quad , \tag{23}$$

then if the iterative procedure adjusts $\underset{\sim}{p}$ to a value near the boundary, J_a approaches infinity because at least one of the g_i terms approaches zero. In this manner the penalty term

$$R \sum_{i=1}^{m} \frac{1}{g_i(\underset{\sim}{p})}$$

tends to keep the values of $\underset{\sim}{p}$ within the admissible region. If R is allowed to become very small, and the constrained minimum lies on the boundary, the value of $\underset{\sim}{p}$ found by minimizing J_a will approach the true minimum point $\underset{\sim}{p}^*$.

To illustrate the Fiacco-McCormick procedure, the following example is presented. The function to be minimized is

$$J(p) = p \tag{24}$$

and the constraints are

$$1 \leq p \leq 2 \ . \tag{25}$$

Putting this constraint in the form of equation (21) gives

$$g_1(p) = p-1 \geq 0$$

$$g_2(p) = 2-p \geq 0 \ , \tag{26}$$

hence

$$J_a = p + R\left[\frac{1}{p-1} + \frac{1}{2-p}\right] \ . \tag{27}$$

Plots of J_a as a function of p for two values of R are shown in Figure 6.

Generally, the minimization procedure is performed several times with decreasing values of R. It can be shown (reference 10) that doing this generates a sequence of points which converges to $\underset{\sim}{p}^*$ (provided that J_a meets certain mathematical requirements).

In combining the Fletcher-Powell and Fiacco-McCormick procedures with CALAHAN it has been assumed that the constraints are of the form given in equation (6), thus,

$$J_a(\underset{\sim}{p},\mathbf{R}) = J(\underset{\sim}{p}) + R \sum_{j=1}^{r} \left[\frac{1}{p_j - a_j} + \frac{1}{-p_j + b_j}\right] \ , \tag{28}$$

and

$$\frac{\partial J_a}{\partial p_j}\left(\underset{\sim}{p}^{(i)},R\right) = \frac{\partial J}{\partial p_j}\left(\underset{\sim}{p}^{(i)}\right) + R\left[\frac{-1}{(p_j^{(i)} - a_j)^2} + \frac{1}{(-p_j^{(i)} + b_j)^2}\right] , \tag{29}$$

where p_j is the jth component of the vector $\underset{\sim}{p}$.

The Fiacco-McCormick approach has been combined with the Naval Postgraduate School Computer Facility's version of the Fletcher-Powell algorithm (subroutine FLAP) and CALAHAN. Several modifications

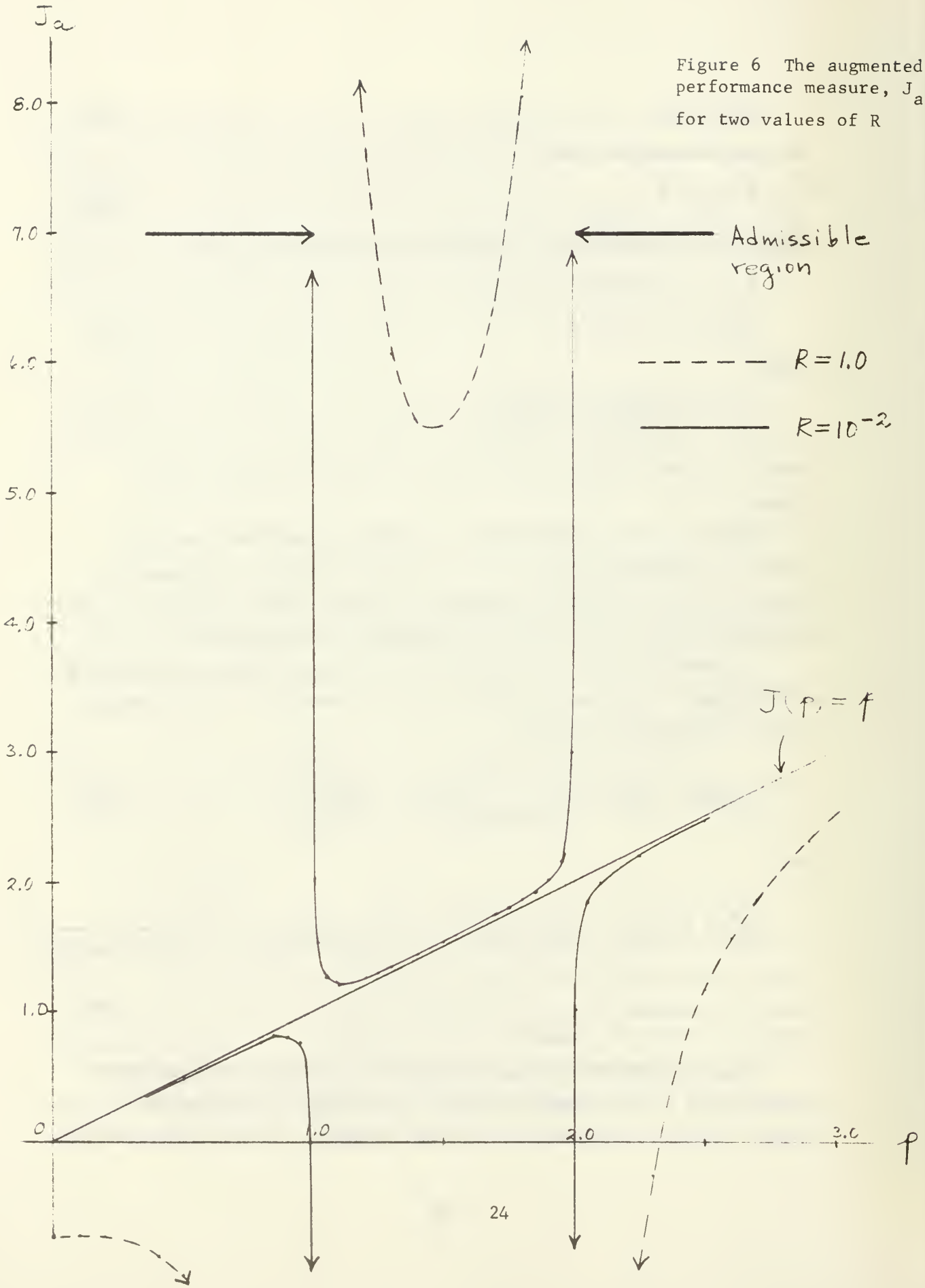


Figure 6 The augmented performance measure, J_a, for two values of R

have been made to improve the operation of the combined program. One of these modifications ensures that $\underset{\sim}{p}$ is never adjusted to lie outside of the feasible region. Such an adjustment can occur in the Fletcher-Powell method during the extrapolation phase of the procedure.

6. ILLUSTRATIVE EXAMPLES

In this section two numerical examples will be presented. The three optimization techniques discussed previously were applied to these examples to provide a basis for comparing the algorithms. Although these examples have networks with only passive elements, the programs are also capable of optimizing linear networks containing active elements. References 7 and 9 present several additional example problems, including some with non-ideal network elements.

Example 1. The problem is to determine the reactive element values in the network shown in Figure 7 so that the frequency response

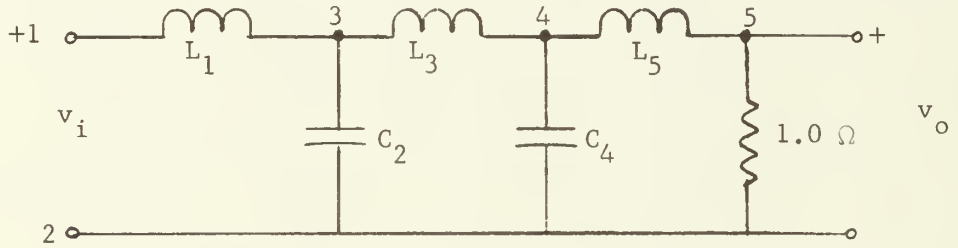


Figure 7. Fifth-order low pass filter

minimizes the performance measure

$$J = \sum_{i=1}^{10} [20 \log_{10}|G_D(f_i)| - 20 \log_{10}|G(f_i)|]^2 \quad . \tag{30}$$

$G(f) = V_{52}(f)/V_{12}(f)$ is the actual frequency response and the desired frequency response characteristic is given by

$$G_D(f_i) = \frac{}{\sqrt{1 + (2\pi f_i)^{2N}}} \tag{31}$$

where N = 5. This desired response is the Butterworth response characteristic for a fifth-order filter; if the element values in the network shown in Figure 7 are

$L_1 = 1.545$ H $C_2 = 1.694$ F $L_3 = 1.3820$ H $C_4 = 0.8944$ F

$L_5 = 0.3090$ H the desired response characteristic will be obtained exactly. The frequencies used were $f_1 = .15$ Hz., .16 Hz., ... , .24 Hz.

The results obtained are summarized in Table 2. Although the

		Pattern Search		Gradient Projection		Fletcher-Powell	
	Fifth-Order Butterworth Value	Initial Value	Final Value	Initial Value	Final Value	Initial Value	Final Value
L_1	1.545	1.600	1.540	1.000	1.557	1.600	1.475
C_2	1.694	2.000	1.700	2.400	1.478	2.000	1.800
L_3	1.382	1.500	1.380	1.100	1.695	1.500	1.312
C_4	0.894	0.900	0.895	0.900	1.064	0.900	0.877
L_5	0.309	0.500	0.307	1.000	0.500	0.500	0.242

Table 2 Element values for fifth-order low pass filter

element values obtained by the pattern search and Fletcher-Powell algorithms differ significantly from the Butterworth values, the frequency response curves obtained are almost idéntical with the desired frequency response curve. The frequency response results obtained from the gradient projection method are shown in Figure 8.

The Fletcher-Powell method took eighteen iteration cycles to determine the values given in Table 2. [An iteration cycle is defined as a minimization of J for a specified value of the factor R in Equation (22)]. If the algorithm operated as anticipated, one iteration cycle should have been sufficient to obtain convergence. The slowness of convergence is attributed to the relative insensitivity of the performance measure to variations in the parameter values; this means that near the minimum the gradients calculated by the perturbation method are likely to be too inaccurate to obtain the expected convergence properties of the algorithm.

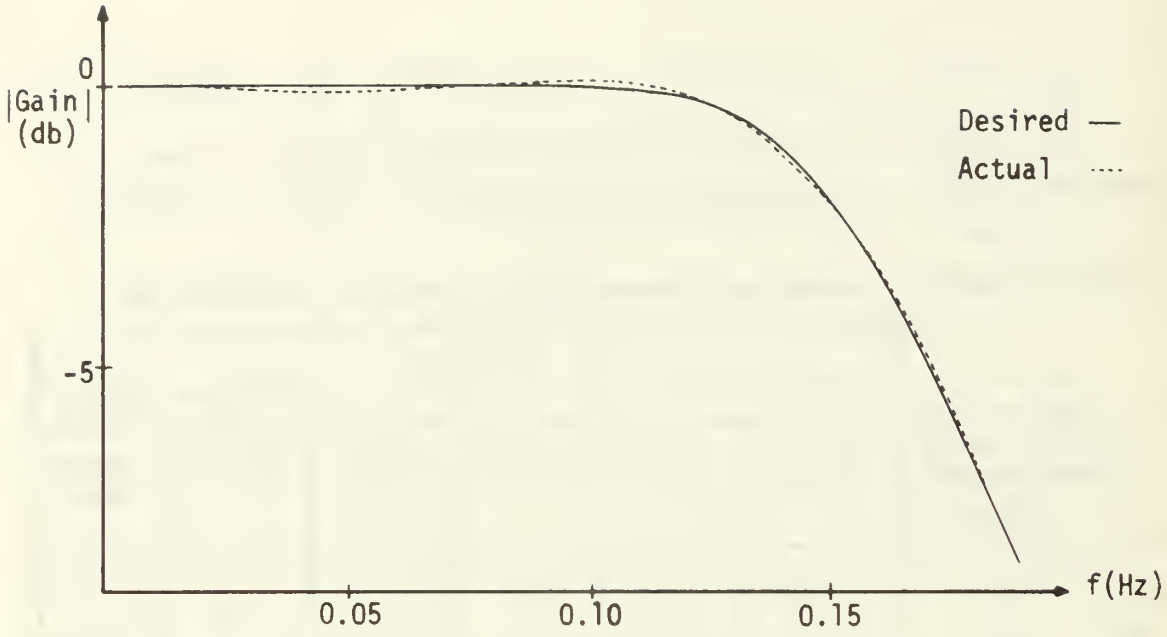


Figure 8 Actual and desired frequency responses for low pass filter

Example 2. The reactive element values in the network shown in Figure 9

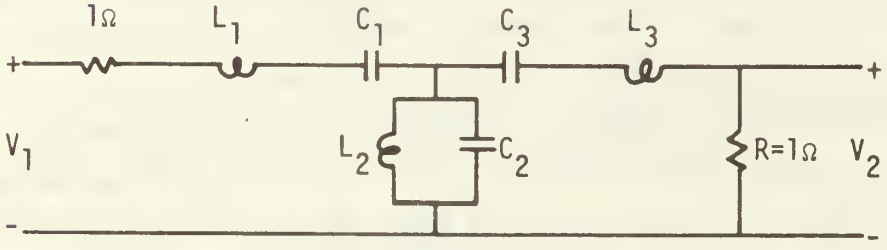


Figure 9 Bandpass filter configuration

are to be determined to minimize the performance measure

$$J = \sum_{i=1}^{q} [20 \log_{10}|G_D(f_i)| - 20 \log_{10}|G(f_i)|]^2$$

where $G(f) = V_2(f)/V_1(f)$ and $20 \log_{10}|G_D(f_i)|$ is the bandpass filter response shown in Figure 10; a tabulation of the gain values on the

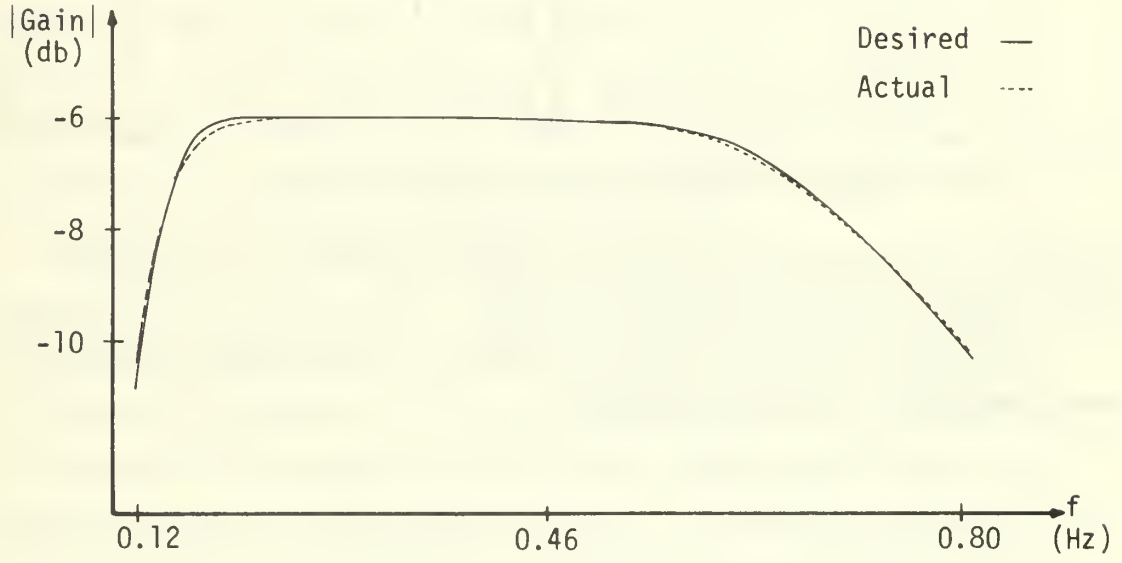


Figure 10 Actual and desired bandpass filter frequency responses

desired frequency response curve is contained in reference 9.

The element values obtained by the three algorithms are contained in Table 3.

The performance of Fletcher-Powell method was less successful than in Example 1. The values in Table 3 were obtained in one iteration cycle, but correspond to a performance measure of J = 175.8.

	Pattern Search		Gradient Projection		Fletcher-Powell	
	Initial Value	Final Value	Initial Value	Final Value	Initial Value	Final Value
L_1	0.500	0.278	1.000	0.231	0.500	0.193
C_1	1.200	0.968	1.200	1.140	1.200	1.213
L_2	0.750	0.512	1.000	0.520	0.500	0.433
C_2	0.750	0.500	1.000	0.485	1.200	0.988
L_3	0.500	0.231	1.000	0.237	0.750	0.050
C_3	1.200	1.070	1.200	1.140	0.750	0.737

Table 3 Bandpass filter element values

By contrast, the performance measure which corresponds to the pattern search values is of the order of 1×10^{-6}.

The frequency response curve obtained for the gradient projection element values is shown in Figure 10.

7. CONCLUSIONS

Three optimization techniques have been combined with the network analysis program CALAHAN and used to solve several illustrative examples. Although the examples contained only networks with passive elements, the programs are capable of optimizing linear networks with active elements. Of the three methods pattern search seems to be most effective; however, it is believed that this conclusion is mainly due to the finite difference approximation used for the gradient in the other two algorithms. The finite difference approximation is also believed to be responsible for the difficulties encountered with the Fletcher-Powell algorithm.

To improve the efficiency of the computational procedures several modifications could be made. First, the network analysis program should be altered so that once the appropriate network trees are calculated, they should be stored instead of being re-computed each time the analysis program is called; this change should reduce the required computation times significantly. Second, the gradient vector should be calculated by using formulas rather than by using a finite difference approximation. This modification should make the gradient projection and Fletcher-Powell algorithms far more efficient. Finally, investigation of techniques for altering the network configuration as well as its element values should broaden the class of problems for which these computer-aided design methods are useful.

REFERENCES

1. Proceedings of the IEEE, Vol. 55, No. 11, Nov. 1967.

2. Calahan, D. A., Computer-Aided Network Design, New York: McGraw-Hill, Inc., 1968.

3. Kuo, F. F. and W. G. Magnuson, Jr., Editors, Computer Oriented Circuit Design, Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1969.

4. IEEE Transaction on Education, Vol. E-12, Nos. 3 and 4, September and December 1969.

5. Rosen, J. B., "The Gradient Projection Method for Nonlinear Programming. Part I. Linear Constraints," J. SIAM (1960), 181-217.

6. Kirk, D. E., Optimal Control Theory: An Introduction. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., to be published 1971.

7. Henry, C. A., "Computer-Aided Design of Passive Filters in the Frequency Domain using Gradient-Projection Minimization Techniques," M. S. Thesis, Naval Postgraduate School, 1969.

8. Wilde, D. J., and C. S. Beightler, Foundations of Optimization. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1967.

9. Lau, J., Computer-Aided Network Design by Optimization in the Frequency Domain, M. S. Thesis, Naval Postgraduate School, December, 1969.

10. Fletcher, R. and M. J. D. Powell, "A Rapidly Convergent Descent Method for Minimization", Comp. J., 6, 2(1963), 163-168.

11. Fiacco, A. V. and G. P. McCormick, "The Sequential Unconstrained Minimization Technique for Nonlinear Programming, A Primal-Dual Method", Management Science, 10, 2(1964), 360-366.

APPENDIX

THE FLETCHER-POWELL OPTIMIZATION PROGRAM

```
FILE: CRTF     FORTRAN  P1                                  CAMBRIDGE  MONITOR  SY

      IMPLICIT REAL*8 (A-H),REAL*8 (O-Z)
      REAL*4 ELT,SLTA
C     WM IS THE WEIGHTING MATRIX, DR THE DESIRED RESPONSE, AL THE LOWER
C     CONSTRAINT LIMIT, AU THE UPPER CONSTRAINT LIMIT, AND THE REST OF
C     THE ARRAYS CONTAIN INFORMATION USED IN CALAHAN.
C
C     NPTS IS THE NUMBER OF FREQUENCY POINTS, JW AND JD ARE INDICATORS,
C     N IS THE NUMBER OF VARIABLE ELEMENTS, NPVA IS THE NUMBER OF
C     PASSIVE VARIABLE ELEMENTS
      DIMENSION WM(100),DR(200),AL(25),AU(25),MP(100,3),
     1ELT(100),MAP(20,5),SLTA(20),VAL(100),VALA(20),C(50),Y11(60),
     2Y12(60),Y(60),Z(60),Y21(60),Y22(60),VALL(50),ZZ(60,2),PP(60,2)
      DIMENSION X(25)
      READ(4,100) NPTS,JW,JD,N,NPVA
  100 FORMAT(5I3)
      IF(JW.LT.0) GO TO 200
C     IF JW.GE.0 READ WEIGHTING MATRIX(WM), OTHERWISE GENERATE WM
      READ(4,101) (WM(I),I=1,NPTS)
  101 FORMAT(F12.4)
      GO TO 201
  200 DO 700 I=1,NPTS
  700 WM(I) = 1.
  201 CONTINUE
      WRITE(6,850)
  850 FORMAT(2X'ERROR',14X,'DESIRED RESP (DR)',/)
      IF(JD.LT.0) GO TO 202
C     IF JD.GE.0 READ DESIRED RESP(DR), OTHERWISE GENERATE DR
      READ(4,101) (DR(I),I=1,NPTS)
      GO TO 203
  202 DF = .01
      DO 701 I=1,NPTS
      FI = I-1
      FR = .15 + FI*DF
      OM = 6.2831*FR*FR
      AD = 1./SQRT(1.+(OM)**(2*N))
      DR(I) = 20.*DLOG10(AD)
  701 WRITE(6,801) FR,DR(I)
  801 FORMAT(2F18.8)
  203 CONTINUE
      READ(4,104) (AL(I),I=1,N),(AU(I),I=1,N),DX
C     AL(I) IS LOWER LIMIT FOR ITH VARIABLE
C     AU(I) IS UPPER LIMIT FOR ITH VARIABLE
C     DX IS PERTURBATION SIZE(IN DECIMAL FRACTION) FOR VARIABLE
C     PARAMETERS
      WRITE(6,799)
  799 FORMAT(2X'THE LOWER LIMITS, UPPER LIMITS, AND DX ARE',/)
      WRITE(6,800) (AL(I),I=1,N)
      WRITE(6,800) (AU(I),I=1,N),DX
  800 FORMAT(9(2XE10.8))
  104 FORMAT(8F10.0)
      NS = 1
161     NSAVE=1
C      READ,PRINT RLC ELEMENTS
C     THE ELEMENTS MUST BE READ IN THE FOLLOWING ORDER-- PASSIVE
C     VARIABLE ELEMENTS, PASIVE FIXED ELEMENTS, ACTIVE VARIABLE
```

```
FILE: ORTF     FORTRAN 21                                    CAMBRIDGE MONITOR SY
C       ELEMENTS, ACTIVE FIXED ELEMENTS
  363   READ(4,1) NPL, NAL, NN, JI,KI,JO,KO,KEY1,
     1       (MP(J,1),MP(J,2),ELT(J),VAL(J),J=1,NPL)
    1  FORMAT(8(I2,1X)/(I2,1X,I2,1X,A1,1X,E10.0))
       READ(4,105) LIN,NOM,OMGMIN,OMGMAX
  105  FORMAT(I1,1X,I3/2E10.0)
       READ(4,106) MODE,IDEB,NEE,EPS
  106  FORMAT(2(I1,1X),I4,E10.2)
       KEY2 = 2
       NVAR = 0
       WRITE(6,71)
71     FORMAT(//,10X,19HCIRCUIT INPUT DATA        ,//)
       WRITE(6,72) NPL,NAL,NN,JI,KI,JO,KO,KEY1,
     1       (MP(J,1),MP(J,2),ELT(J),VAL(J),J=1,NPL)
72     FORMAT(8X,8(I2,1X)/(8X,I2,1X,I2,1X,A1,1X,E18.9))
C      IF ACTIVE ELEMENTS,READ AND PRINT
       IF(NAL)3,2,3
    3  READ(4,4) ((MAP(J,I),I=1,4),ELTA(J),VALA(J),J=1,NAL)
    4  FORMAT(4(I2,1X),A1,1X,E10.0)
       WRITE(6,73)((MAP(J,I),I=1,4),ELTA(J),VALA(J),J=1,NAL)
73     FORMAT(8X,4(I2,1X),A1,1X,E18.9)
2      KEY3=1
C      NAVA IS THE NUMBER OF ACTIVE VARIABLE ELEMENTS
       IW=1
       NAVA = N-NPVA
       NEVAL = 0
       EOLD = 1.D23
       STEP = 1.
       READ(4,102) R,DV
  102  FORMAT(F20.8)
       WRITE(6,103) R,DV
  103  FORMAT(5X'R = 'D18.8,5X'DV = ' D18.8,/)
       DO 10 I=1,NPVA
   10  X(I) = VAL(I)
       IF(NPVA.EQ.N) GO TO 12
       DO 11 I=1,NAVA
   11  X(I+NPVA) = VALA(I)
   12  CONTINUE
   30  CALL FLAP(N,X,F,MODE,IDEB,IFLAG,NEE,MEE,R,JEL,
     1NPL,NAL,NN,JI,KI,JO,KO,KEY2,MP,ELT,VAL,MAP,ELTA,VALA,Y11,Y12,VALL,
     2JW,NVAR,KEY3,Y,NP,J11,Y21,Y22,Y7,N7,J22,JP,Y7,J7,PE,LIN,NOM,
     3OMGMIN,OMGMAX,KEY1,DX,AL,AL,WW,DR,NPVA,NAVA,NPTS,EPS)
       FNEW = F
C      WRITE(6,1199) F,(X(I),I=1,N)
 1199  FORMAT(9(1XE14.5))
C      WRITE(6,1200) (X(I),I=1,N)
 1200  FORMAT(10X,'THE COORDINATES ARE',/,8(1XD14.5),//)
       WRITE(6,1201) F,IFLAG,NEE
 1201  FORMAT(2X'THE MIN IS ',D18.8,'IFLAG IS',I4,'NEE IS ',I4,/)
       R = R/DV
       NEVAL = NEVAL + NEE
       IF(FNEW.LE.EOLD) GO TO 20
       FBEST = EOLD
       GO TO 32
   20  DN = DABS((EOLD-FNEW)/EOLD)
```

```
FILE: CPTF     FORTRAN  P1                                  CAMBRIDGE  MONITOR  SY
      IF(DN.LE..001) GO TO 28
      FOLD = FNEW
      GO TO 20
   28 FBEST = FNEW
 1220 FORMAT(2X'THE TOTAL NO. OF FCN. EVAL. IS ',I4,'FBEST = ',E18.8,/)
      DO 30 I=1,NPVA
   30 VAL(I) = X(I)
      IF(NPVA.EQ.N) GO TO 32
      DO 31 I=1,NAVA
   31 VALA(I) = X(I+NPVA)
   32 CONTINUE
      WRITE(6,1220) NEVAL,FBEST
      WRITE(6,1106) (VAL(J),J=1,NPL)
 1106 FORMAT(5X'THE PASSIVE ELEMENT VALUES ARE',/,6(2XE18.8))
      IF(NAL.EQ.0) GO TO 33
      WRITE(6,1107) (VALA(J),J=1,NAL)
 1107 FORMAT(5X'THE ACTIVE ELEMENT VALUES ARE',/,6(2XE18.8))
   33 CONTINUE
      END
```

```
FIL': OPTF2    FORTRAN  BE                                  CAMBRIDGE  MONITOR  SY'

      SUBROUTINE DOFS(N,X,G,E,R,
     1NPL,NAL,NN,JI,KI,JO,KO,KEY2,MP,FLT,VAL,MAP,FLTA,VALA,Y11,Y12,VALL,
     2JW,NVAR,KEY3,Y,NP,J11,Y21,Y22,Z,NZ,J22,IF,ZZ,J7,PP,LIN,NOM,
     3OMGMIN,OMGMAX,KEY1,DX,AL,AU,WM,DR,NPVA,NAVA,NPTS)
      IMPLICIT REAL*8 (A-H),REAL*8(O-Z)
      REAL*4 FLT,FLTA
      DIMENSION X(1),G(1),MP(100,3),FLT(1),VAL(1),MAP(20,5),
     1FLTA(1),VALA(1),Y11(1),Y12(1),VALL(1),Y(1),Y21(1),Y22(1),Z(1),
     2ZZ(60,2),PP(60,2),AL(1),AU(1),WM(1),DR(1)
      DIMENSION D(50),DSQ(50),R2(100)
      NT2 = N*2
      DO 1 I=1,N
      D(I) = X(I)-AL(I)
    1 D(N+I) = -X(I) +AU(I)
      DO 4 I=1,NT2
    4 DSQ(I) = 1./D(I)**2
C     SET VARIABLE ELEMENTS EQUAL TO X(I)
      DO 5  I=1,NPVA
    5 VAL(I) = X(I)
      IF(NPVA.EQ.N) GO TO 3
      DO 2 I =1,NAVA
    2 VALA(I) = X(I+NPVA)
    3 CONTINUE
      NP1 = N+1
      DO 430 JK=1,NP1
      IPERT = JK-1
      IF(IPERT.EQ.0) GO TO 52
      IF(IPERT.GT.NPVA) GO TO 31
      TMP = VAL(IPERT)
      VAL(IPERT) = VAL(IPERT)*(1.+DX)
      GO TO 52
   31 NJP = IPERT-NPVA
      TMP = VALA(NJP)
      VALA(NJP) = VALA(NJP)+(1.+DX)
C     CALCULATE COEFFICIENTS OF NUMERATOR,DENOMINATOR
   52 GO TO (6,7,6,8,5,6),KEY1
    6 CALL TOPOL(NPL,NAL,NN,JI,KI,JO,KO,2,KEY2,MP,FLT,VAL,MAP,FLTA,
     1VALA,Y11,Y12,VALL,JW,NVAR,KEY3,Y,NP,J11)
      GO TO 9
    7 CALL TOPOL(NPL,NAL,NN,JI,KI,JO,KO,2,KEY2,MP,FLT,VAL,MAP,FLTA,
     1VALA,Y11,Y12,VALL,JW,NVAR,KEY3,Y,NP,J11)
      GO TO 9
    8 CALL TOPOL(NPL,NAL,NN,JO,KO,JI,KI,2,KEY2,MP,FLT,VAL,MAP,FLTA,
     1VALA,Y11,Y12,VALL,JW,NVAR,KEY3,Y,NP,J11)
    9 GO TO (10,11,11,12,12,13),KEY1
   10 CALL TOPOL(NPL,NAL,NN,JI,KI,JO,KO,2,KEY2,MP,FLT,VAL,MAP,FLTA,
     1VALA,Y21,Y22,VALL,JW,NVAR,KEY3,Z,NZ,J22)
      GO TO 14
   11 CALL TOPOL(NPL,NAL,NN,JI,KI,JO,KO,1,KEY2,MP,FLT,VAL,MAP,FLTA,
     1VALA,Y21,Y22,VALL,JW,NVAR,KEY3,Z,NZ,J22)
      GO TO 14
   12 CALL TOPOL(NPL,NAL,NN,JI,KI,JO,KO,4,KEY2,MP,FLT,VAL,MAP,FLTA,
     1VALA,Y21,Y22,VALL,JW,NVAR,KEY3,Z,NZ,J22)
      GO TO 14
   13 CALL TOPOL(NPL,NAL,NN,JO,KO,JI,KI,2,KEY2,MP,FLT,VAL,MAP,FLTA,
```

```
FILE: CPTF2     FORTRAN  P5                                  CAMBRIDGE  MONITOR  SY

     1VALA,Y21,Y22,VALL,IW,NVAR,KEY3,Z,NZ,J22)
C      REMOVE COMMON LOWER DEGREE ZERO COEFFICIENTS OF NUMERATOR,DENOM.
   15  CONTINUE
       DO 20 J=1,40
       IF(Y(1))16,15,16
15     IF(Z(1))16,17,16
16     JP=NP-J+1
       JZ=NZ-J+1
       GO TO 21
17     DO 18 K=1,NP
18     Y(K)=Y(K+1)
       DO 19 K=1,NZ
19     Z(K)=Z(K+1)
20     CONTINUE
C      CALCULATE DEGREE OF NUMERATOR,DENOMINATOR,IGNORING HIGHER DEGREE
C      ZERO COEFFICIENTS
   21  CONTINUE
       DO 30 J=1,40
       JJ=JP-J+1
       IF(Y(JJ))32,30,32
32     JP=JP-J+1
       GO TO 34
30     CONTINUE
34     DO 35 J=1,40
       JJ=JZ-J+1
       IF(Z(JJ))33,35,33
33     JZ=JZ-J+1
       GO TO 36
35     CONTINUE
C      CALCULATE ZEROS
   36  CALL MULLER(Y,JP,ZZ)
C      CALCULATE POLES
       CALL MULLER(Z,JZ,PP)
C      CALCULATE FREQUENCY RESPONSE
       CALL FREQQ(LIN,NOM,OMGMIN,OMGMAX,JP,JZ,Y,Z,KEY1,P2)
  444  CONTINUE
C     END OF CALC OF FREQ RESPONSE
C
C
C
C      WRITE(6,902) (P2(I),I=1,NPTS)
  902  FORMAT(5X'E RES DATA',/(5(1XE1'.6,/)))
       IF(IPERT.NE.0) GO TO 26
       E = 0.
       DO 25 I=1,NPTS
   25  E = E + WM(I)*(DP(I)-P2(I))**2
C      WRITE(6,903) E
  903  FORMAT(2X'E =',E14.6,/)
       GO TO 430
   26  DE = -E
       DO 27 I=1,NPTS
   27  DE = DE+WM(I)*(DP(I)-P2(I))**2
       G(IPERT) = DE/(DX*TMP)
       IF(IPERT.GT.NPVA) GO TO 432
       VAL(IPERT) = TMP
```

```
FILE: DSTF2    FORTRAN  DS                                  CAMBRIDGE  MONITOR  SY

      GO TO  30
  432 VALA(NIP) = TMP
  430 CONTINUE
      DO 404  I=1,N
  404 G(I) = G(I) -R*(DSO(I)-DSO(N+I))
      RETURN
      END
C
C
      SUBROUTINE CHECK(X,AL,AU,N,JEL)
      IMPLICIT REAL*8 (A-H),REAL*8(O-Z)
      REAL*4 FLT,FLTA
      DIMENSION X(1),AL(1),AU(1)
      JEL = 1
      DO 1 I=1,N
      IF(X(I).GE.AL(I).AND.X(I).LE.AU(I)) GO TO 1
      JEL = 0
      RETURN
    1 CONTINUE
      RETURN
      END
C
C
      SUBROUTINE FUNCT(N,X,F,G,P,ARG,L,MODE,IDER,IFLAG,NFE,MFE,R,JEL,
     1NPL,MAL,NN,JT,KI,JJ,KO,KEY2,MP,FLT,VAL,MAP,FLTA,VALA,Y11,Y12,VALI,
     2JW,NVAR,KEY3,Y,NP,J11,Y21,Y22,Z,N7,J22,JP,Z7,J7,PP,LIN,NOM,
     3OMGMIN,OMGMAX,KEY1,DX,AL,AU,MM,DR,NPVA,NAVA,NOTS)
      IMPLICIT REAL*8 (A-H),REAL*8(O-Z)
      REAL*4 FLT,FLTA
      DIMENSION X(1),G(1),P(1),MP(100,3),FLT(1),VAL(1),MAP(20,5),
     1FLTA(1),VALA(1),Y11(1),Y12(1),VALI(1),Y(1),Y21(1),Y22(1),Z(1),
     2ZZ(60,2),PP(40,2),AL(1),AU(1),WM(1),DR(1)
      DIMENSION XTMP(25),NCON(25),IDENT(6)
      COMMON HARG1(25)
      DATA IDENT/4H    ,4HREGN,4HEXTR,4HINTR,4H    ,4HMINI/
C
C     IFLAG INDICATES CURRENT STATUS OF FUNCTION EVALUATION
C           0= NO MINIMUM FOUND CONTINUE EXPLORATION
C           1= MINIMUM NOT FOUND WITHIN MAXIMUM NUMBER OF EVALUATIONS
C              SPECIFIED BY ITRS PARAMETER
C          -1= MINIMUM FOUND WITHIN VALUE OF ITRS

      GO TO (6,5,5,5,3),L
    5 IF(L.NE.2) GO TO 8
      DO 17 I = 1,N
   17 XTMP(I) = X(I)
    8 DO 16 IR=1,N
   16 X(IR)=X(IR)+ARG*P(IR)
      NFE = NFE+1
      IF(NFE.GE.MFE) IFLAG=1
      GO TO 4
    6 IFLAG=0
      MD=MODE-2
```

```
    4 IF(L.NE.2) GO TO 7
      CALL CHECK(X,AL,AU,N,JFL)
      IF(JFL.NE.0) GO TO 7
      DO 10 I =1,N
   10 X(I) = XTMP(I)
      IF(JFL.EQ.0) RETURN
    7 CALL PDER(N,X,G,E,R,
     1MPL,NAL,NN,JT,KI,JO,KO,KEY2,MP,FLT,VAL,MAP,FLTA,VALA,Y11,Y12,VALL,
     2JW,NVAR,KEY3,Y,NP,J11,Y21,Y22,Z,N7,J22,JP,Z7,JZ,PP,LIN,NOM,
     3OMGMIN,OMGMAX,KEY1,DX,AL,AU,WM,DR,NPVA,NAVA,NPTS)
    8 IF(MF) 101,105,102
  101 IF(MODE.EQ.0) RETURN
      IF (L.EQ.1.OR.L.EQ.5) GO TO 102
      RETURN
  105 IF(MOD(L,2).NE.1) RETURN
  102 WRITE(6,1000) IDENT(L+1),E,(X(I),I=1,N)
10000FORMAT(/2X,A4,2X/6H FUNCTION,1PX,20HINDEPENDENT VARIABLE//1X,E14.7
     1,3X,7E14.7/(24X,7E14.7))
      IF(IDER.NE.1) RETURN
      WRITE(6,1001) (G(I),I=1,N)
 1001 FORMAT(23X/23 X17H GRADIENT VECTORS// (24X,7E14.7))
  100 RETURN
      END
C
C
C
      SUBROUTINE FLAP(N,X,E,MODE,IDER,IFLAG,NEE,MEE,R,JFL,
     1MPL,NAL,NN,JT,KI,JO,KO,KEY2,MP,FLT,VAL,MAP,FLTA,VALA,Y11,Y12,VALL,
     2JW,NVAR,KEY3,Y,NP,J11,Y21,Y22,Z,N7,J22,JP,Z7,JZ,PP,LIN,NOM,
     3OMGMIN,OMGMAX,KEY1,DX,AL,AU,WM,DR,NPVA,NAVA,NPTS,EPS)
      IMPLICIT REAL*8 (A-H),REAL*8(O-Z)
      REAL*4 FLT,FLTA
C     NOTE THAT G(1), G(1) ARE NOT IN THE FOLLOWING STATEMENT
      DIMENSION X(1),MP(100,3),FLT(1),VAL(1),MAP(20,5),
     1FLTA(1),VALA(1),Y11(1),Y12(1),VALL(1),Y(1),Y21(1),Y22(1),Z(1),
     2ZZ(60,2),PP(60,2),AL(1),AU(1),WM(1),DR(1)
      DIMENSION H(325),S(25),G(25),SIGMA(25),GAMMA(25)
      COMMON HARC1(25)
      IF(MODE) 50,54,70
   70 IF(MODE-3) 53,52,50
   50 MODE=2
   52 IF(IDER) 55,54,71
   71 IF(IDER-1) 54,54,55
   55 IDER=1
   54 IF(MEE.EQ.0) MEE=200
      NEE=0
      K=1
      DO 200 I=1,N
      NMI=N-I
      H(K)=1.
      IF(NMI.EQ.0) GO TO 101
      DO 201 J=1,NMI
      IN=K+J
  201 H(IN)=0.
```

```
FILE: CPTF2    FORTRAN  F5                                  CAMBRIDGE  MONITOR  SY

  200 K=K+N-I+1
  101 CLOF=1.E+8
      ICOUNT=0
      CALL FUNCT(N,X,F,G,S,ARG,1,MODE,IOER,IFLAG,NFE,MEE,R,JEL,
     1NPL,NAL,NN,JI,KI,JO,KO,KEY2,MP,ELT,VAL,MAP,ELTA,VALA,Y11,Y12,VALL,
     2JW,NVAR,KEY3,Y,NP,J11,Y21,Y22,Z,N7,J22,JP,Z7,J7,PP,LIN,NOM,
     3OMGMIN,OMGMAX,KEY1,DX,AL,AU,WM,DR,NRVA,NAVA,NPTS)
      IF(IFLAG.NE.0) GO TO 2
  102 IF(CLDF.LE.F) GO TO 2
      ICOUNT=ICOUNT+1
      STEP = 1.
      DO 301 I=1,N
      SIGMA(I)=X(I)
      GAMMA(I)=G(I)
  301 S(I)=-UPDOT(H,G,I,N,KD)
      FB=F
      GB=DOT(G,S,N)
      IF(GB.GE.0) GO TO 2
      FIT=STEP
      CLDF=F
   21 FA=FB
      GA=GB
   87 CALL FUNCT(N,X,F,G,S,FIT,2,MODE,IOER,IFLAG,NFE,MEE,R,JEL,
     1NPL,NAL,NN,JI,KI,JO,KO,KEY2,MP,ELT,VAL,MAP,ELTA,VALA,Y11,Y12,VALL,
     2JW,NVAR,KEY3,Y,NP,J11,Y21,Y22,Z,N7,J22,JP,Z7,J7,PP,LIN,NOM,
     3OMGMIN,OMGMAX,KEY1,DX,AL,AU,WM,DR,NRVA,NAVA,NPTS)
      IF(IFLAG.NE.0) GO TO 84
      FIT = FIT/4.
      STEP = STEP/4.
      GO TO 87
   84 IF(IFLAG.NE.0) GO TO 2
      FB=F
      GB=DOT(G,S,N)
      IF(GB.GE.0) GO TO 19
      IF(FB.GE.FA) GO TO 19
      FIT=4.*FIT
      STEP=4.*STEP
      GO TO 21
   19 NINT = 0
C     IF INTERP. FAILS 5 CONSECUTIVE TIMES, PROCESS TERM.
   22 ZE=3.*(FA-FB)/FIT+GA+GB
      WSQ = ZE**2-GA*GB
      IF(WSQ.LT.0.) GO TO 80
      W = DSQRT(WSQ)
      GO TO 81
   80 W = 0.
   81 CONTINUE
      IF((GA+ZE).LT.0.) GO TO 82
      FLAM = FIT*(1.-(GA+ZE+W)/(GA+GB+2.*ZE))
      GO TO 83
   82 FLAM = FIT*(1.-(GA/(GA+ZE-W)))
   83 CONTINUE
      ARG=-FLAM
      CALL FUNCT(N,X,F,G,S,ARG,3,MODE,IOER,IFLAG,NFE,MEE,R,JEL,
     1NPL,NAL,NN,JI,KI,JO,KO,KEY2,MP,ELT,VAL,MAP,ELTA,VALA,Y11,Y12,VALL,
```

```
FILE: OPTF2     FORTRAN  B5                                  CAMBRIDGE  MONITOR  SYS

     2JW,NVAR,KEY3,Y,NP,J11,Y21,Y22,Z,N7,J22,JP,Z7,J7,PP,LIN,NOM,
     3OMGMIN,OMGMAX,KEY1,DX,AL,AU,WM,DR,NPVA,NAVA,NPTS)
      IF(IFLAG.NE.0) GO TO 2
      IF(F.LE.FA.AND.F.LE.FB) GO TO 401
      STEP=STEP/4.
      IF(FB.LT.FA) GO TO 500
      NINT = NINT+1
      IF(NINT.GE.5) GO TO 2
      GO TO 501
  500 NINT = 0
C     ARG 1 IN CALL FUNCT NEXT STATEMENT HAS BEEN CHANGED FROM 4 TO 3
C     WRITE(6,900)
      CALL FUNCT(N,X,F,G,S,ELAM,3,MODE,IDER,IFLAG,NEE,MEE,R,JEL,
     1NEL,NAL,NN,JI,KI,JO,KO,KEY2,MP,ELT,VAL,MAP,ELTA,VALA,Y11,Y12,VALL,
     2JW,NVAR,KEY3,Y,NP,J11,Y21,Y22,Z,N7,J22,JP,Z7,J7,PP,LIN,NOM,
     3OMGMIN,OMGMAX,KEY1,DX,AL,AU,WM,DR,NPVA,NAVA,NPTS)
      IF(IFLAG.NE.0) GO TO 2
      GO TO 401
  501 GB=DOT(G,S,N)
      IF(GB.GE.0) GO TO 13
      IF(ICOUNT.GT.N.AND.STEP.LT.1.E-6) GO TO 2
   13 FB=F
      ELT=ELT-ELAM
      GO TO 22
  401 DO 600 I=1,N
      SIGMA(I)=X(I)-SIGMA(I)
  600 GAMMA(I)=G(I)-GAMMA(I)
      SG=DOT(SIGMA,GAMMA,N)
      SS=DOT(S,S,N)
      GG=DOT(SIGMA,SIGMA,N)
      IF(ICOUNT.LT.N) GO TO 701
      IF(DSQRT(SS).LT.EPS.AND.DSQRT(GG).LT.EPS) GO TO 2
  701 DO 10 I=1,N
   10 S(I)=UPDOT(H,GAMMA,I,N,KD)
      GHG=DOT(S,GAMMA,N)
      KK=1
      IF(SG.LE.EPS.OR.GHG.LE.EPS) GO TO 2
      DO 800 I=1,N
      DO 800 J=I,N
      H(KK)=H(KK)+SIGMA(I)*SIGMA(J)/SG-S(I)*S(J)/GHG
  800 KK=KK+1
      IF(NEE.LT.MEE) GO TO 102
    2 CALL FUNCT(N,X,F,G,S,0.5,MODE,IDER,IFLAG,NEE,MEE,R,JEL,
     1NEL,NAL,NN,JI,KI,JO,KO,KEY2,MP,ELT,VAL,MAP,ELTA,VALA,Y11,Y12,VALL,
     2JW,NVAR,KEY3,Y,NP,J11,Y21,Y22,Z,N7,J22,JP,Z7,J7,PP,LIN,NOM,
     3OMGMIN,OMGMAX,KEY1,DX,AL,AU,WM,DR,NPVA,NAVA,NPTS)
      RETURN
      END
C
C
C
      FUNCTION DOT(A,B,N)
      IMPLICIT REAL*8 (A-H),REAL*8(O-Z)
      REAL*4 ELT,ELTA
      DIMENSION A(1),B(1)
```

```
FILE: DOTF2     FORTRAN EF                                  CAMBRIDGE  MONITOR  SY
      DOT = 0.
      DO 10 I=1,N
   10 DOT=DOT+A(I)*B(I)
      RETURN
      END
C
C
C
      FUNCTION UPDOT(A,B,I,N,KD)
      IMPLICIT REAL*8 (A-H),REAL*8(O-Z)
      REAL*4 ELT,ELTA
      DIMENSION A(1),B(1)
      K=I
      UPDOT = 0.
      IM1 = I-1
      IF(IM1.LE.0) GO TO 30
      DO 10 J=1,IM1
      UPDOT = UPDOT + A(K)*B(J)
   10 K=K+N-J
   30 DO 20 J=1,N
      IN=K+J-I
   20 UPDOT = UPDOT +A(IN)*B(J)
      RETURN
      END
```

DISTRIBUTION LIST

	No. of copies
Mr. G. H. S. Williamson Jet Propulsion Laboratory 4800 Oak Grove Drive Pasadena, California 91103	10
Defense Documentation Center Cameron Station, Bldg. 5 Alexandria, Virginia 22314	20
Library Naval Postgraduate School Monterey, California 93940	2
Professor Donald E. Kirk Naval Postgraduate School Monterey, California 93940	5
Professor C. E. Menneken Dean of Research Administration Naval Postgraduate School Monterey, California 93940	2

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Naval Postgraduate School Monterey, California 93940	2a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED 2b. GROUP

3. REPORT TITLE

Computer-Aided Design of Linear Networks in the Frequency Domain

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

Technical Report, 1970

5. AUTHOR(S) (First name, middle initial, last name)

Donald E. Kirk

6. REPORT DATE 11 June 1970	7a. TOTAL NO. OF PAGES 46	7b. NO. OF REFS 11
8a. CONTRACT OR GRANT NO. b. PROJECT NO. c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) NPS-52KI0061A 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	

10. DISTRIBUTION STATEMENT

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING [redacted] ACTIVITY Jet Propulsion Laboratory 4800 Oak Grove Drive Pasadena, California 91103

13. ABSTRACT

The design of linear networks to satisfy frequency domain performance specifications is formulated as a problem in nonlinear programming. Three optimization algorithms, pattern search, gradient projection, and the Fletcher-Powell method, are applied in conjunction with the network analysis program CALAHAN to the solution of the nonlinear programming problem. Examples which illustrate the range of application of the design programs are presented.

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Computer-aided circuit design						
Optimization						
Gradient projection						
Pattern search						
Fletcher-Powell						